Predicting Diverse Subsets Using Structural SVMs

Yisong Yue, Thorsten Joachims

Cornell University
Department of Computer Science

Diversified Retrieval

- **Ambiguous queries:**
  - Example query: "SVM"
    - ML method
    - Service Master Company
    - Magazine
    - School of veterinary medicine
    - Sport Verein Meppen e.V.
    - SVM software
    - SVM books
  - "submodular" performance measure

    ➔ make sure each user gets at least one relevant result
- **Learning Queries:**
  - Find all information about a topic
  - Eliminate redundant information

Query: SVM

1. Kernel Machines
2. SVM book
3. SVM-light
4.
5.
6.
7.

Query: SVM

1. Kernel Machines
2. Service Master Co
3. SV Meppen
4. UArizona Vet. Med.
5. SVM-light
6. Intro to SVM
7. …

[YueJo08]

Generic Structural SVM

- **Application Specific Design of Model**
  - Loss function $\Delta(y_i, y)$
  - Representation $\Phi(x, y)$
- **Prediction:**

$$\hat{y} = argmax_{y \in Y} \{\vec{w}^T \Phi(x,y)\}$$

- **Training:**

$$\begin{aligned} \min_{\vec{w},\vec{\xi}\geq 0} \quad & \frac{1}{2}\vec{w}^T\vec{w} + \frac{C}{n}\sum_{i=1}^{n}\xi_i \\ s.t. \quad & \forall y \in Y \backslash y_1 : \vec{w}^T\Phi(x_1, y_1) \geq \vec{w}^T\Phi(x_1, y) + \Delta(y_1,y) - \xi_1 \\ & \ldots \\ & \forall y \in Y \backslash y_n : \vec{w}^T\Phi(x_n, y_n) \geq \vec{w}^T\Phi(x_n, y) + \Delta(y_n,y) - \xi_n \end{aligned}$$

- **Applications:** Parsing, Sequence Alignment, Clustering, etc.

Applying StructSVM to New Problem

- **General**
  - SVM-struct algorithm and implementation
  - Theory (e.g. number of iterations independent of n)
- **Application specific**
  - Loss function $\Delta(y_i, y)$
  - Representation $\Phi(x, y)$
  - Algorithms to compute

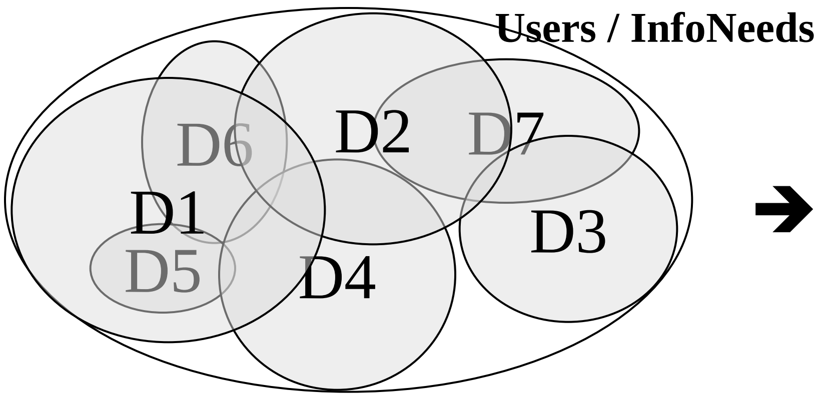
$$\hat{y} = argmax_{y \in Y} \{\vec{w}^T \Phi(x_i, y)\}$$

$$\hat{y} = argmax_{y \in Y} \{\Delta(y_i, y) + \vec{w}^T \Phi(x_i, y)\}$$

- **Properties**
  - General framework for discriminative learning
  - Direct modeling, not reduction to classification/regression
  - “Plug-and-play”

Approach

- **Prediction Problem:**
  - Given set **x**, predict size *k* subset **y** that satisfies most users.
- **Approach: Topic Red. ≈ Word Red. [SwMaKi08]**



➔ **y** = { D1, D2, D3, D4 }

  - Weighted Max Coverage: $\mathbf{y} = \underset{y \subset x, |y|=k}{\mathrm{argmax}} \left\{ \sum_{w \in \bigcup(y)} score(w) \right\}$
  - Greedy algorithm is 1-1/e approximation [Khuller et al 97]

➔ **Learn the benefit weights:** $score(w) = \mathbf{w}^T \phi(w, x)$

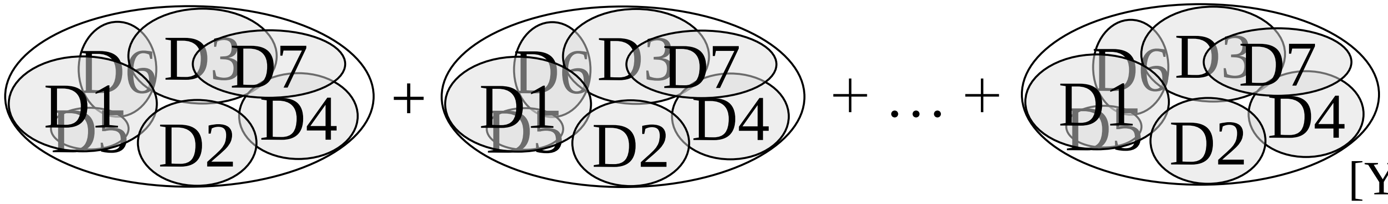
[YueJo08]

Features Describing Word Importance

- **How important is it to cover word w**
  - w occurs in at least X% of the documents in x
  - w occurs in at least X% of the titles of the documents in x
  - w is among the top 3 TFIDF words of X% of the documents in x
  - w is a verb

  → Each defines a feature in $\phi(w, x)$
- **How well a document d covers word w**
  - w occurs in d
  - w occurs at least k times in d
  - w occurs in the title of d
  - w is among the top k TFIDF words in d

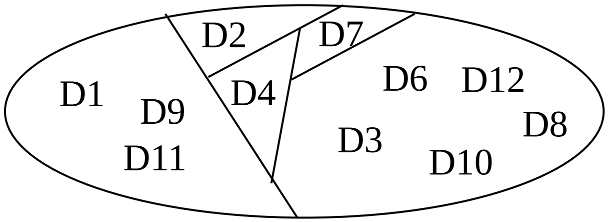
  → Each defines a separate vocabulary and scoring function



[YueJo08]

Loss Function and Separation Oracle

- **Loss function:** $\Delta(y_i, y)$
  - Popularity-weighted percentage of subtopics not covered in **y**
    - → More costly to miss popular topics
  - Example:



$$\Delta(y_i, \{D1, D10\}) = 3/12$$
$$\Delta(y_i, \{D2, D7\}) = 10/12$$

- **Separation oracle:** $\hat{y} = argmax_{y \in Y} \{\Delta(y_i, y) + \vec{w}^T \Phi(x_i, y)\}$
  - Again a weighted max coverage problem
    - → add artificial word for each subtopic with percentage weight
  - Use greedy algorithm again

[YueJo08]

Experiments

- **Data:**
  - TREC 6-8 Interactive Track
  - Relevant documents manually labeled by subtopic
  - 17 queries (~700 documents), 12/4/1 training/validation/test
  - Subset size k=5, two feature sets (div, div2)
- **Results:**

Method	Loss
Random	0.469
Okapi	0.472
Unweighted Model	0.471
Essential Pages	0.434
$\text{SVM}^{\Delta}_{div}$	0.349
$\text{SVM}^{\Delta}_{div2}$	0.382